奈学教育

多线程并发

肖秋平 Eric

自我介绍

肖秋平 Eric

阿吉豆 **IT副总裁 CIO**
阿里巴巴 **架构师/技术专家**
上海惠普HP **技术研发经理**

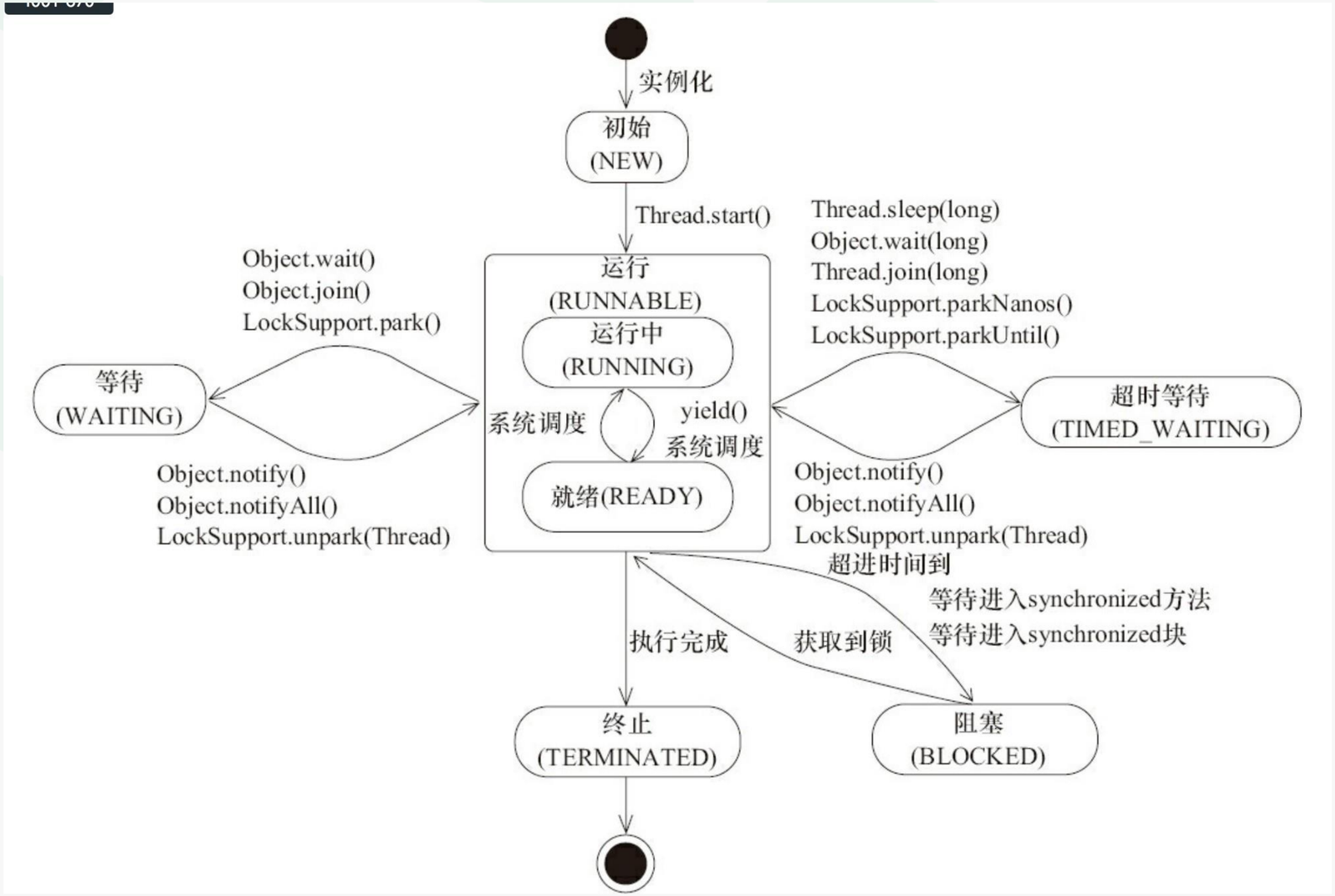
- 14+年大型世界500强外企和IT互联网工作经验，曾参与并主导日活百万级音乐交易类、直播类移动APP的整体分布式架构设计，中台建设；
- 2年带领团队移动互联网+音乐文娱的创业经历；
- 1年上海阿吉豆产品经理、IT副总裁 CIO 经历， 曾负责AJIDOU阿吉豆全国800多家零售连锁门店企业信息化总体战略规划设计，产品技术创新与组织管理；
- 原上海HP惠普PPS-JAVA部门经理带领30人左右团队；
- 资深架构师与独立音乐人综合体，有过创业经历，懂公司运营。

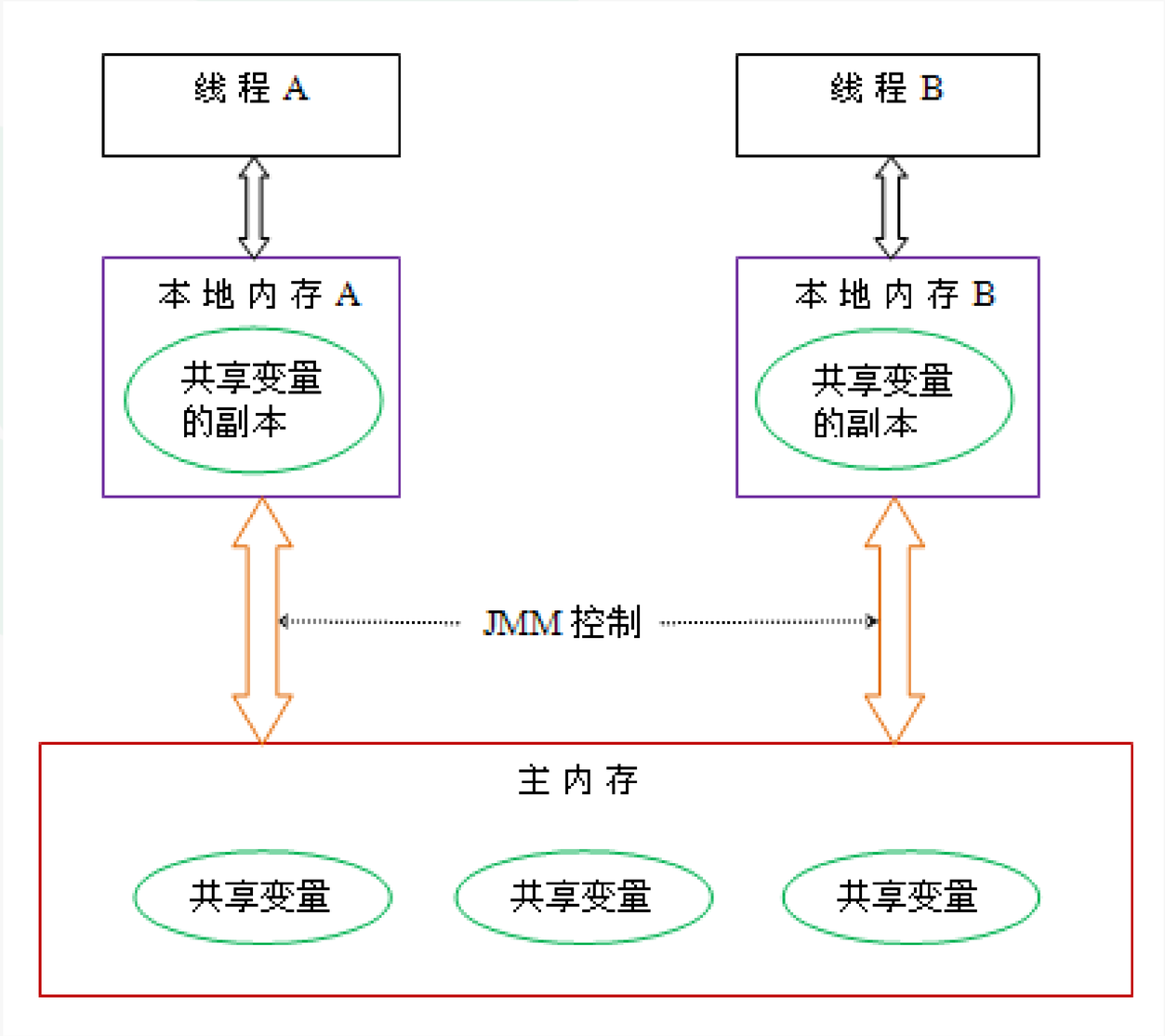
肖秋平 Xiao Qiuping　　Eric

1. Thread.start()做了哪些事情？
2. 多线程有哪些状态？
3. Java Memory Model是什么？
4. Volatile关键字的内存语义和应用场景？
5. Synchronized关键字的内存语义和应用场景？
6. Final关键字的内存语义？
7. J.U.C工具包下支持并发的工具类

1. 进程
2. 线程

线程的状态


实例化
初始
(NEW)
Thread.start()
运行
(RUNNABLE)
运行中
(RUNNING)
Object.wait()
Object.join()
LockSupport.park()
等待
(WAITING)
Object.notify()
Object.notifyAll()
LockSupport.unpark(Thread)
系统调度
yield()
系统调度
就绪(READY)
Thread.sleep(long)
Object.wait(long)
Thread.join(long)
LockSupport.parkNanos()
LockSupport.parkUntil()
超时等待
(TIMED_WAITING)
Object.notify()
Object.notifyAll()
LockSupport.unpark(Thread)
超进时间到
等待进入synchronized方法
等待进入synchronized块
执行完成
获取到锁
终止
(TERMINATED)
阻塞
(BLOCKED)


线程 A
线程 B
本地内存 A
共享变量的副本
本地内存 B
共享变量的副本
JMM 控制
主内存
共享变量
共享变量
共享变量

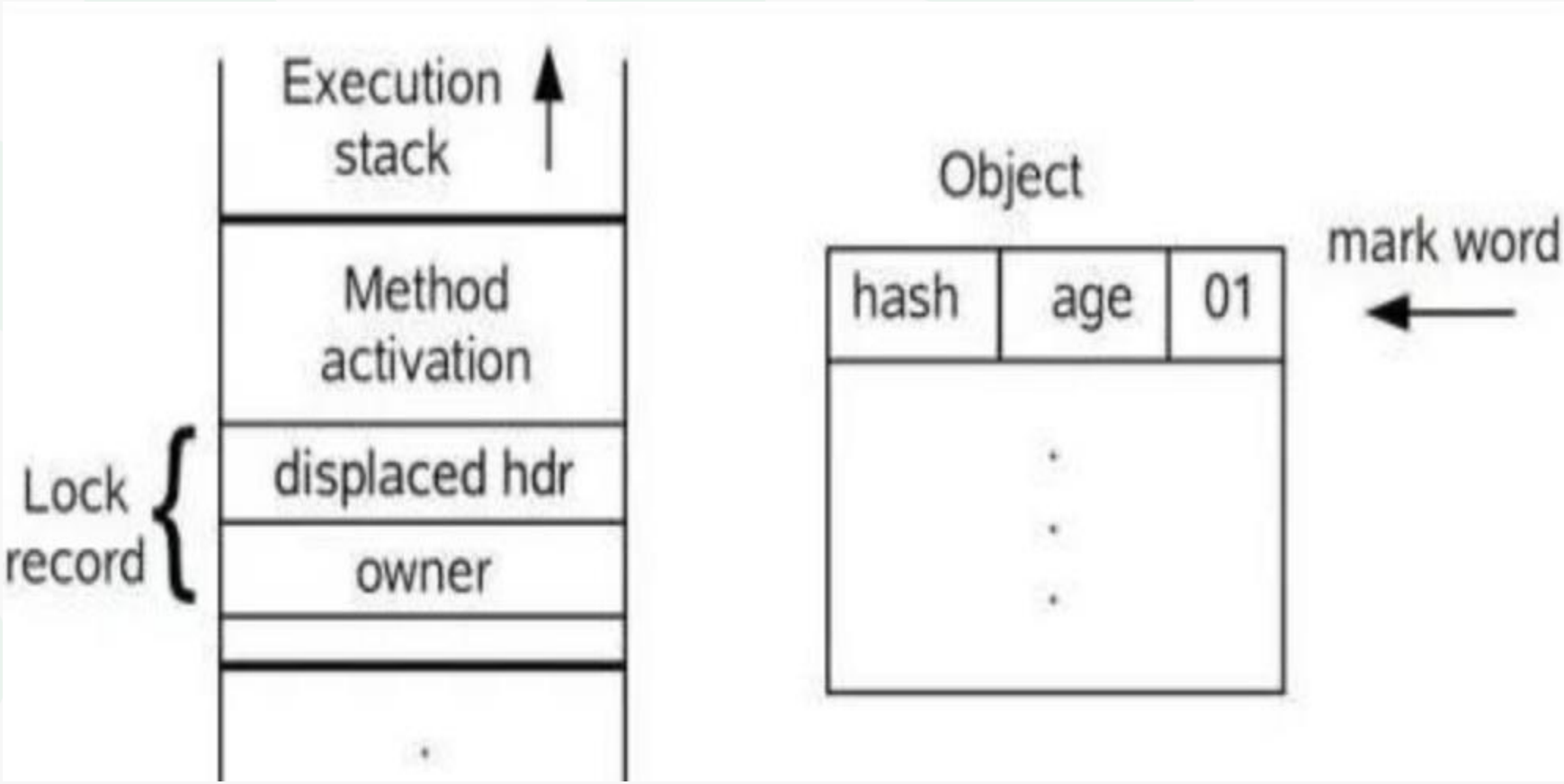
- 实例方法
- 静态方法
- 实例方法中的同步块
- 静态方法中的同步块

长　　度	内　　容	说　　明
32/64bit	Mark Word	存储对象的 hashCode 或锁信息等
32/64bit	Class Metadata Address	存储到对象类型数据的指针
32/32bit	Array length	数组的长度（如果当前对象是数组）

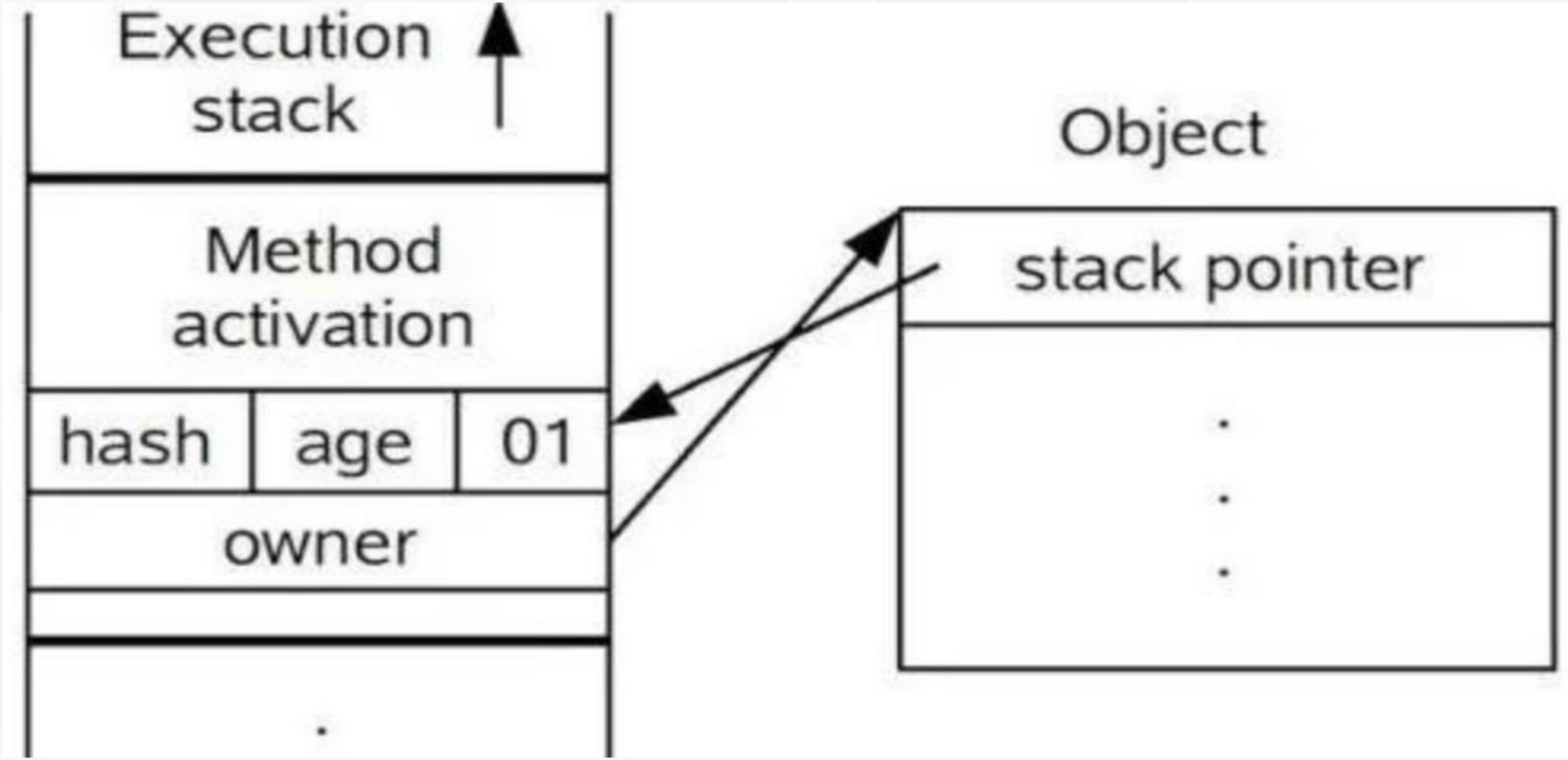
锁状态	25bit	4bit	1bit 是否是偏向锁	2bit 锁标志位
无锁状态	对象的 hashCode	对象分代年龄	0	01

锁状态	25bit		4bit	1bit	2bit
	23bit	2bit		是否是偏向锁	锁标志位
轻量级锁	指向栈中锁记录的指针				00
重量级锁	指向互斥量（重量级锁）的指针				10
GC 标记	空				11
偏向锁	线程 ID	Epoch	对象分代年龄	1	01

- 实例方法
- 静态方法
- 实例方法中的同步块
- 静态方法中的同步块



- 实例方法
- 静态方法
- 实例方法中的同步块
- 静态方法中的同步块



- 可见性（工作内存里的副本失效，从主存里去刷新）
- 禁止指令重排序优化

- 加锁顺序
- 加锁时限
- 避免长事务

- BlockingQueue
- ArrayBlockingQueue DelayQueue LinkedBlockingDeque LinkedBlockingQueue LinkedTransferQueue PriorityBlockingQueue SynchronousQueue

方法\处理方式	抛出异常	返回特殊值	一直阻塞	超时退出
插入方法	add(e)	offer(e)	put(e)	offer(e,time,unit)
移除方法	remove()	poll()	take()	poll(time,unit)
检查方法	element()	peek()	不可用	不可用

- HashMap容量
- 锁Segment(bin)
- Put→ 发生冲突时，元素个数>=8，链表转红黑树，元素个数<6，转回到链表
- 红黑树便于删除、增加新的元素，通过左旋和右旋来平衡二叉树

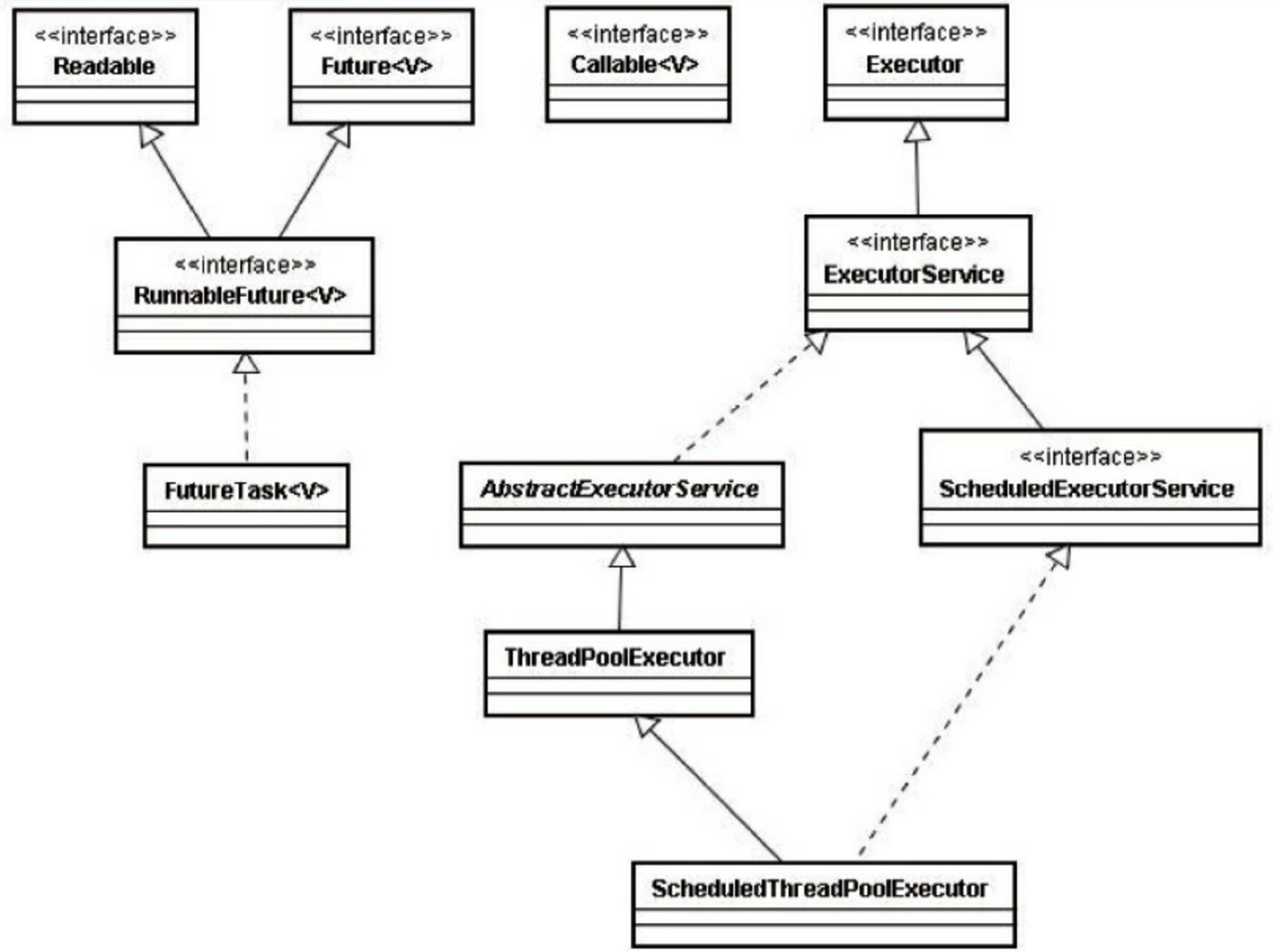
- AtomicBoolean：以原子更新的方式更新boolean；
- AtomicInteger：以原子更新的方式更新Integer；
- AtomicLong：以原子更新的方式更新Long；

- CompareAndSwap(CAS)

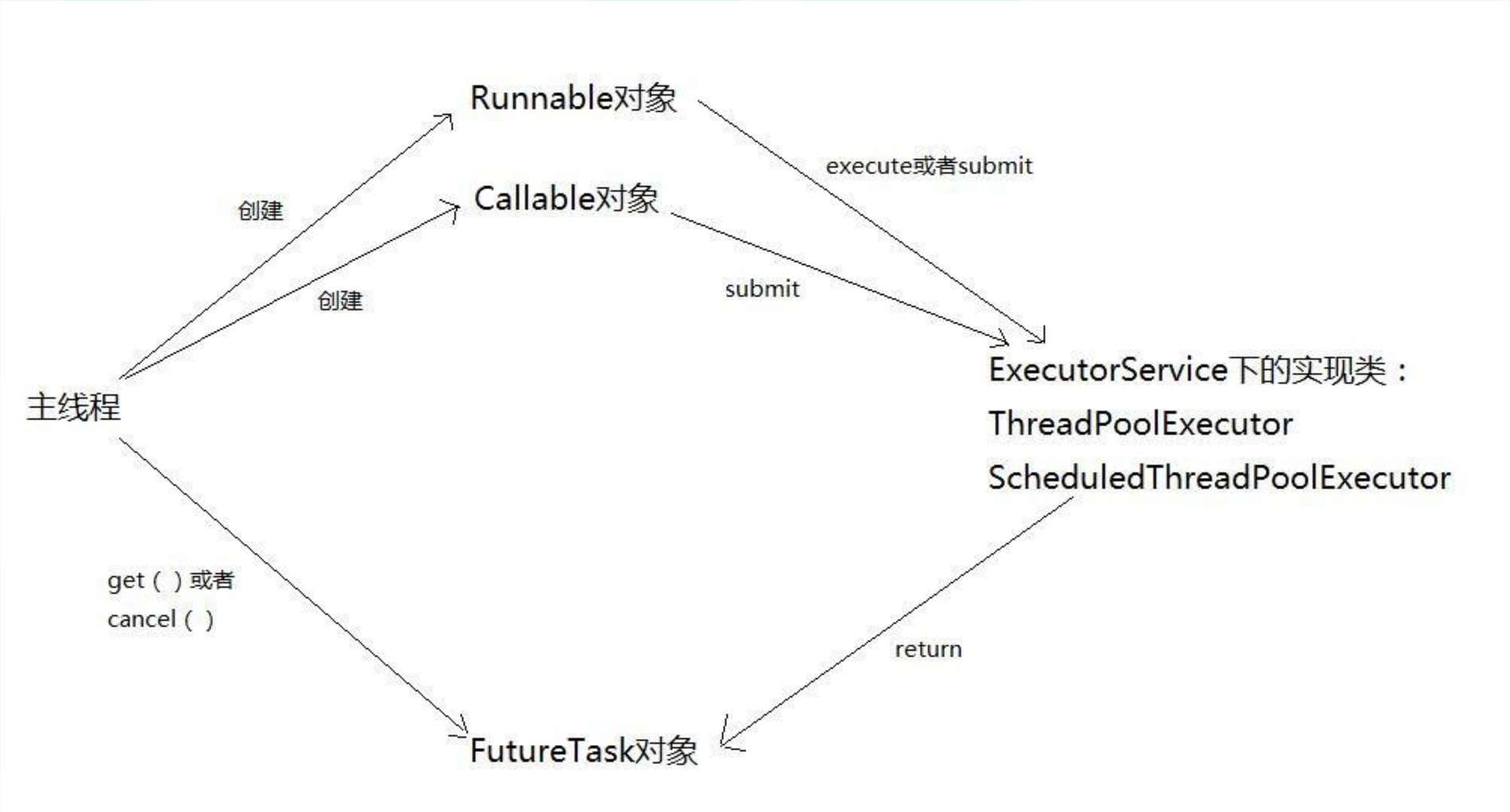
- 可重入锁
- AQS同步器
- FairLock and UnfairLock
- Condition

对比项	Object Monitor Methods	Condition
前置条件	获取对象的锁	调用 Lock.lock() 获取锁 调用 Lock.newCondition() 获取 Condition 对象
调用方式	直接调用 如：object.wait()	直接调用 如：condition.await()
等待队列个数	一个	多个
当前线程释放锁并进入等待状态	支持	支持
当前线程释放锁并进入等待状态，在等待状态中不响应中断	不支持	支持
当前线程释放锁并进入超时等待状态	支持	支持
当前线程释放锁并进入等待状态到将来的某个时间	不支持	支持
唤醒等待队列中的一个线程	支持	支持
唤醒等待队列中的全部线程	支持	支持

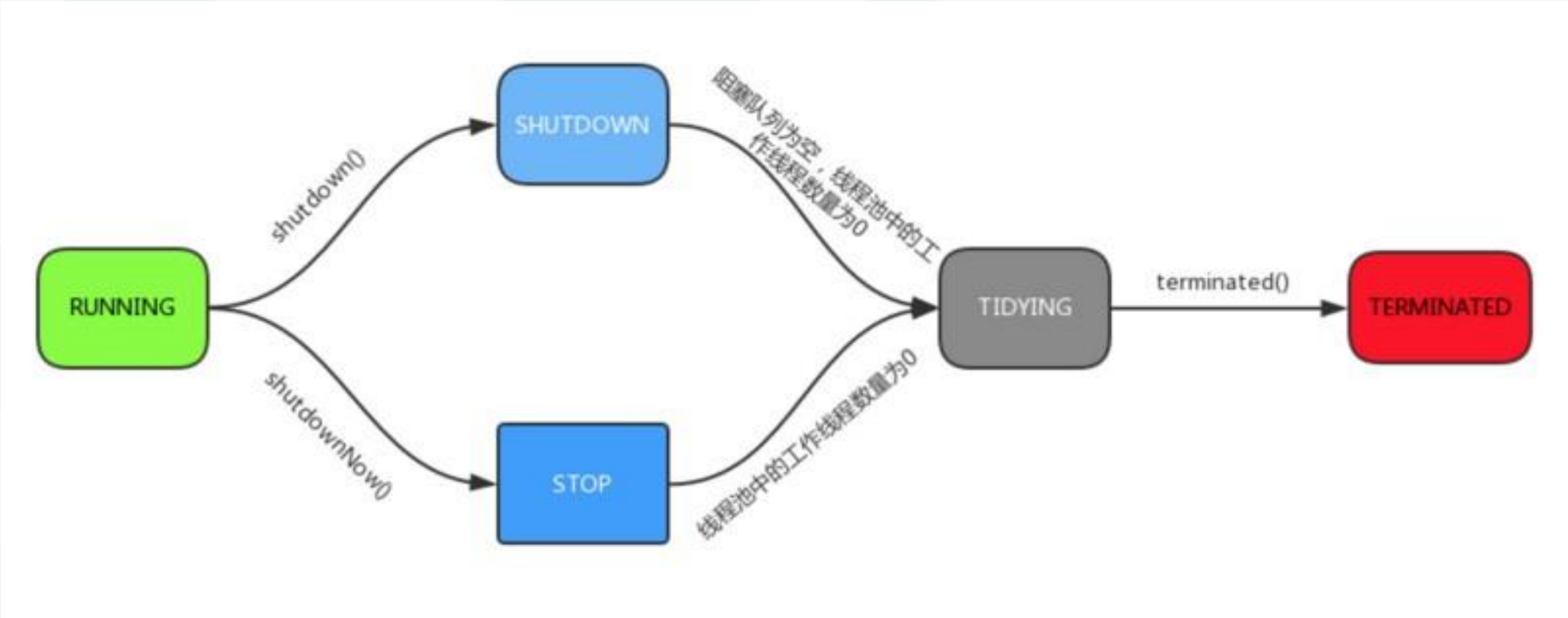
- CountDownLatch
- CyclicBarrier
- Exchanger
- Executors

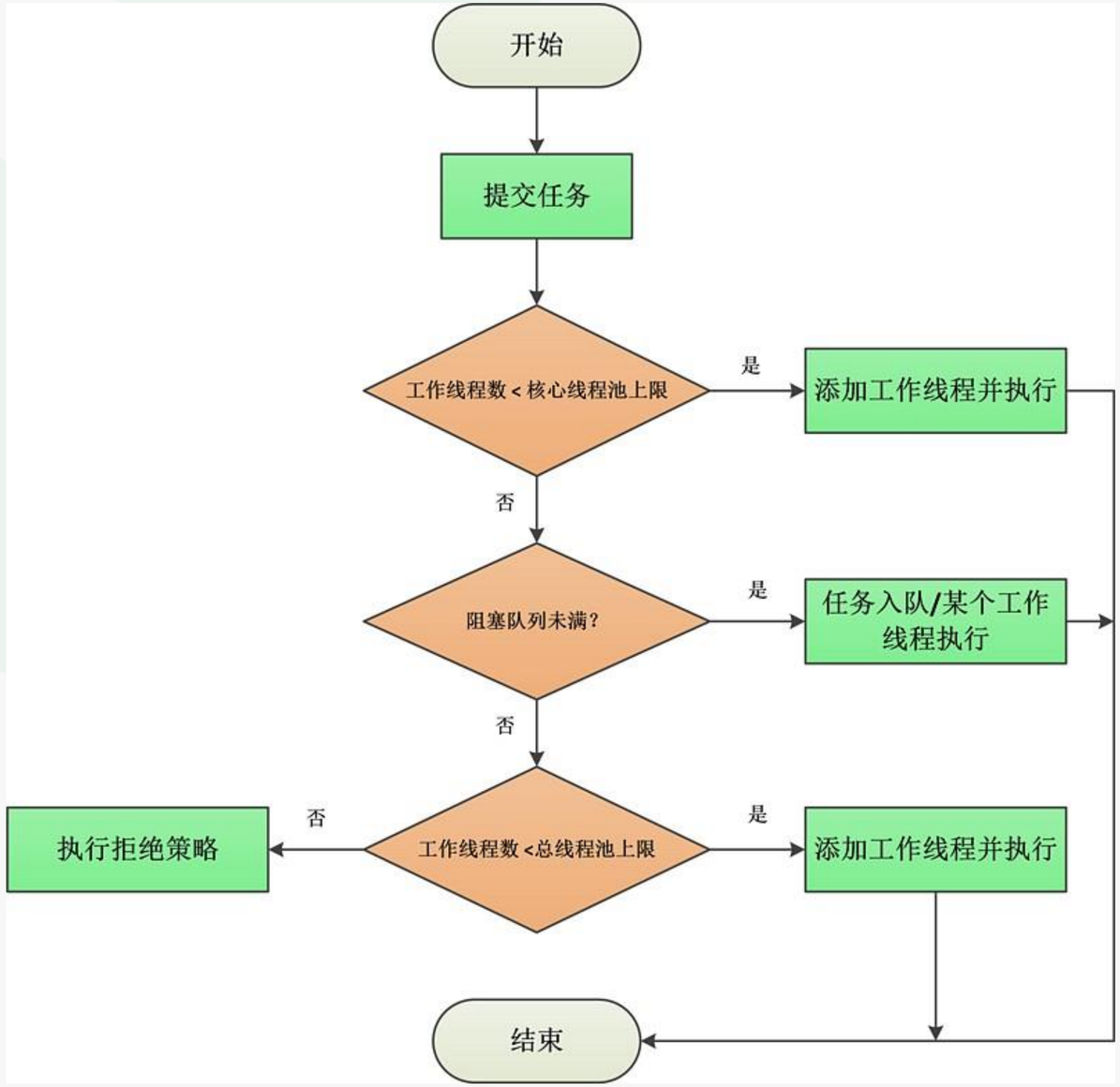


- Executors


Runnable对象
Callable对象
创建
创建
execute或者submit
submit
ExecutorService下的实现类：
ThreadPoolExecutor
ScheduledThreadPoolExecutor
主线程
get ( ) 或者
cancel ( )
return
FutureTask对象

- ThreadPoolExecutor




开始
提交任务
工作线程数 < 核心线程池上限
是
添加工作线程并执行
否
阻塞队列未满?
是
任务入队/某个工作线程执行
否
工作线程数 <总线程池上限
否
执行拒绝策略
是
添加工作线程并执行
结束

奈学教育

让每个人持续提升职业能力